На этом занятии в классе DArray добавим возможность применять операции инкремента и декремента для значений элементов динамического массива.

Очевидно, для этого нужно в классе Item объявить следующие методы:

```language-cpp
    class Item {
    ...
    public:
    ...
        int operator++();
        int operator--();
    };
```

А их реализацию пропишем в файле darray.cpp следующим образом:

```language-cpp
int DArray::Item::operator++()
{
    if(index >= current->length || index < 0)
        return value_error; // операции ++ и -- можно выполнять только с
записанными элементами массива

    current->data[index]++;
    return current->data[index];
}

int DArray::Item::operator--()
{
    if(index >= current->length || index < 0)
        return value_error; // операции ++ и -- можно выполнять только с
записанными элементами массива

    current->data[index]--;
    return current->data[index];
}
```

Здесь снова получается дублирование кода, но это можно поправить так же, как это мы делали с арифметическими операциями.

Итак, после объявления этих методов, ожидается, что мы можем использовать операции инкремента и декремента с элементами массива, например, так:

```language-cpp
#include <iostream>
#include "darray.h"

int main()
{
    DArray ar1, ar2;

    ar1[10] = 100;
    ar1[3] = 5;

    int v1 = ar1[3]++; // ошибка
    int v2 = ++ar1[3]; // ok

    std::cout << v1 << " " << v2 << std::endl;

    for(int i = 0;i < ar1.size(); ++i)
        std::cout << ar1.get_data()[i] << " ";

    return 0;
}
```

Но, как видим, операция инкремента в постфиксной форме выдает ошибку, говоря, что такого метода не существует. Почему так? Дело в том, что мы с вами хорошо знаем, инкремент в префиксной и постфиксной формах должен работать по разному. Следовательно, и методы, которые их реализуют, так же должны быть разными. То, что было объявлено в классе Item, соответствует операциям инкремента и декремента в префиксной форме. Чтобы добавить аналогичные операции для постфиксной формы, было решено воспользоваться механизмом перегрузки методов и объявлять их с одним параметром (в классе Item):

```language-cpp
        int operator++(); // для префиксной формы
        int operator--(); // для префиксной формы
        int operator++(int); // для постфиксной формы
        int operator--(int); // для постфиксной формы
```

Реализации последних двух методов запишем в виде:

```language-cpp
int DArray::Item::operator++(int)
{
    if(index >= current->length || index < 0)
        return value_error; // операции ++ и -- можно выполнять только с
записанными элементами массива

    int value = current->data[index];
```

```
    current->data[index]++;
    return value;

    /*
    или в виде:
    return current->data[index]++;
    */
}

int DArray::Item::operator--(int)
{
    if(index >= current->length || index < 0)
        return value_error; // операции ++ и -- можно выполнять только с
записанными элементами массива

    return current->data[index]--;
}
```

Теперь, мы совершенно свободно можем использовать любую форму операций инкремента и декремента:

```language-cpp
int main()
{
    DArray ar1, ar2;

    ar1[10] = 10;
    ar1[3] = 5;

    int v1 = ar1[3]--;
    int v2 = ++ar1[10];

    std::cout << v1 << " " << ar1[3] << std::endl;
    std::cout << v2 << " " << ar1[10] << std::endl;

    for(int i = 0;i < ar1.size();++i)
        std::cout << ar1.get_data()[i] << " ";

    return 0;
}
```

Дублирование кода, которое получилось при определении этих операций, я предлагаю вам убрать самостоятельно.